

Event-B Course

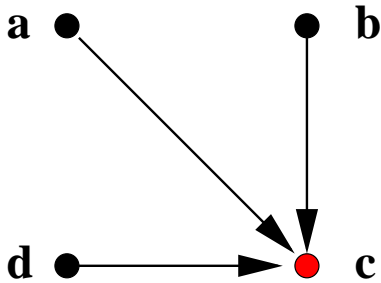
7. Routing Algorithm for Mobile Agent

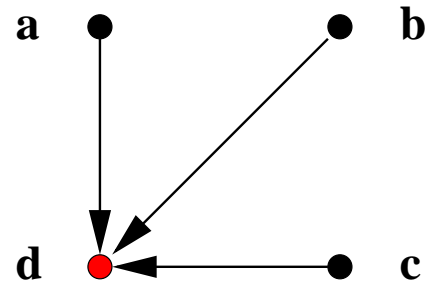
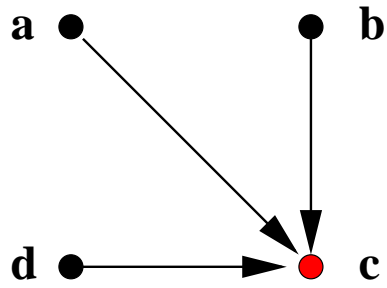
Jean-Raymond Abrial

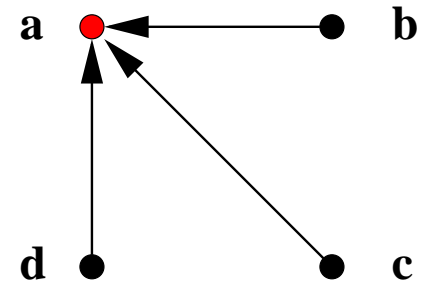
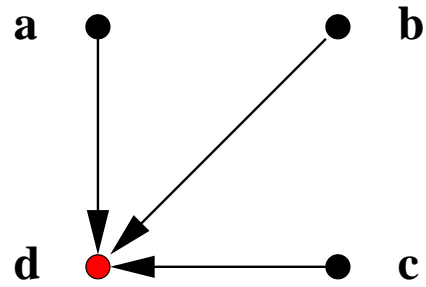
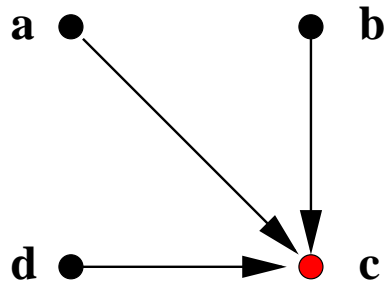
September-October-November 2011

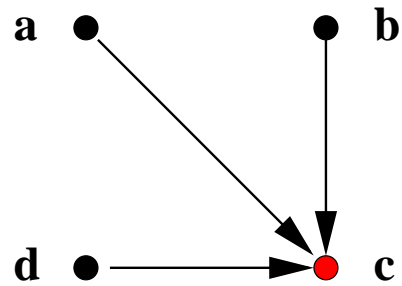
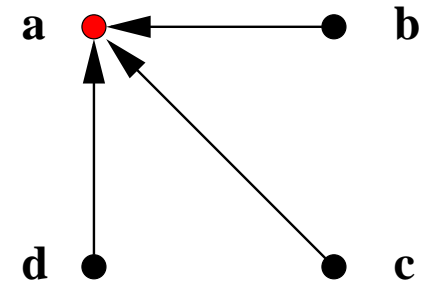
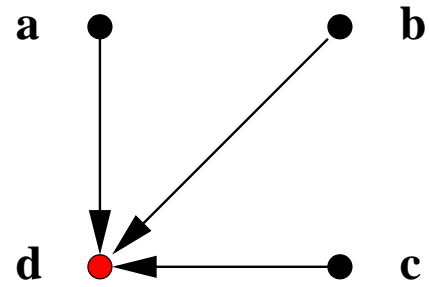
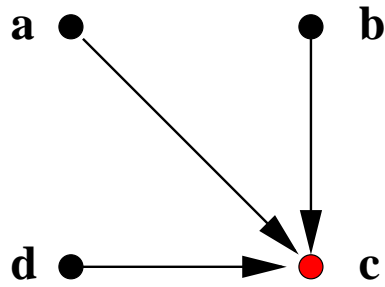
- No more learning about refinement and abstraction (practicing)
- No more learning about modeling conventions (practicing)
- Re-using dynamically the small tree theory we already developed
- Study a practical problem in distributed computing communication
- The example comes from the following paper:
L. Moreau. *Distributed Directory Service and Message Routing for Mobile Agent*. Science of Computer Programming 2001.

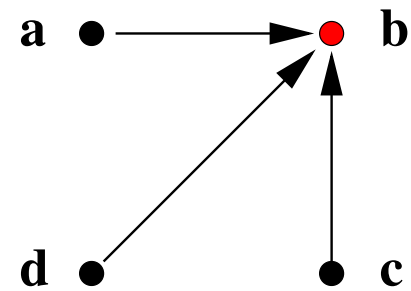
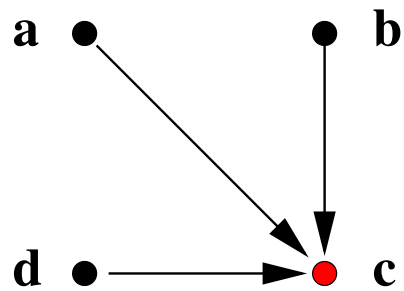
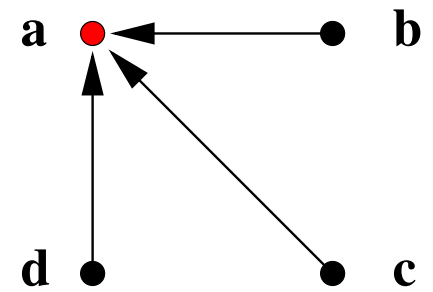
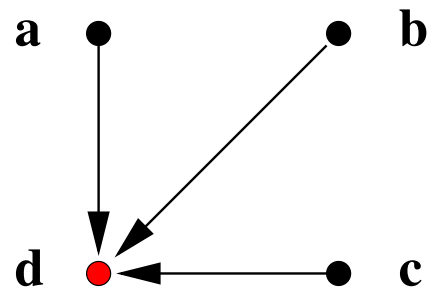
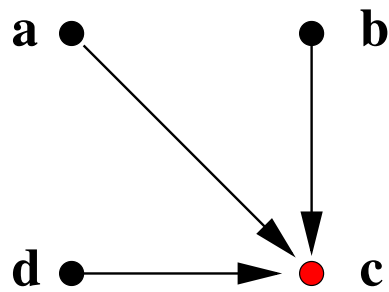
-
- A mobile agent \mathcal{M} is supposed to travel between sites
 - Some fixed agents at sites want to send messages to \mathcal{M}
 - In an abstract world:
 - the moves of \mathcal{M} are instantaneous
 - the traveling of messages between sites takes no time
 - the knowledge of the moves of \mathcal{M} is also instantaneous
 - Thus fixed agents always send messages where \mathcal{M} is





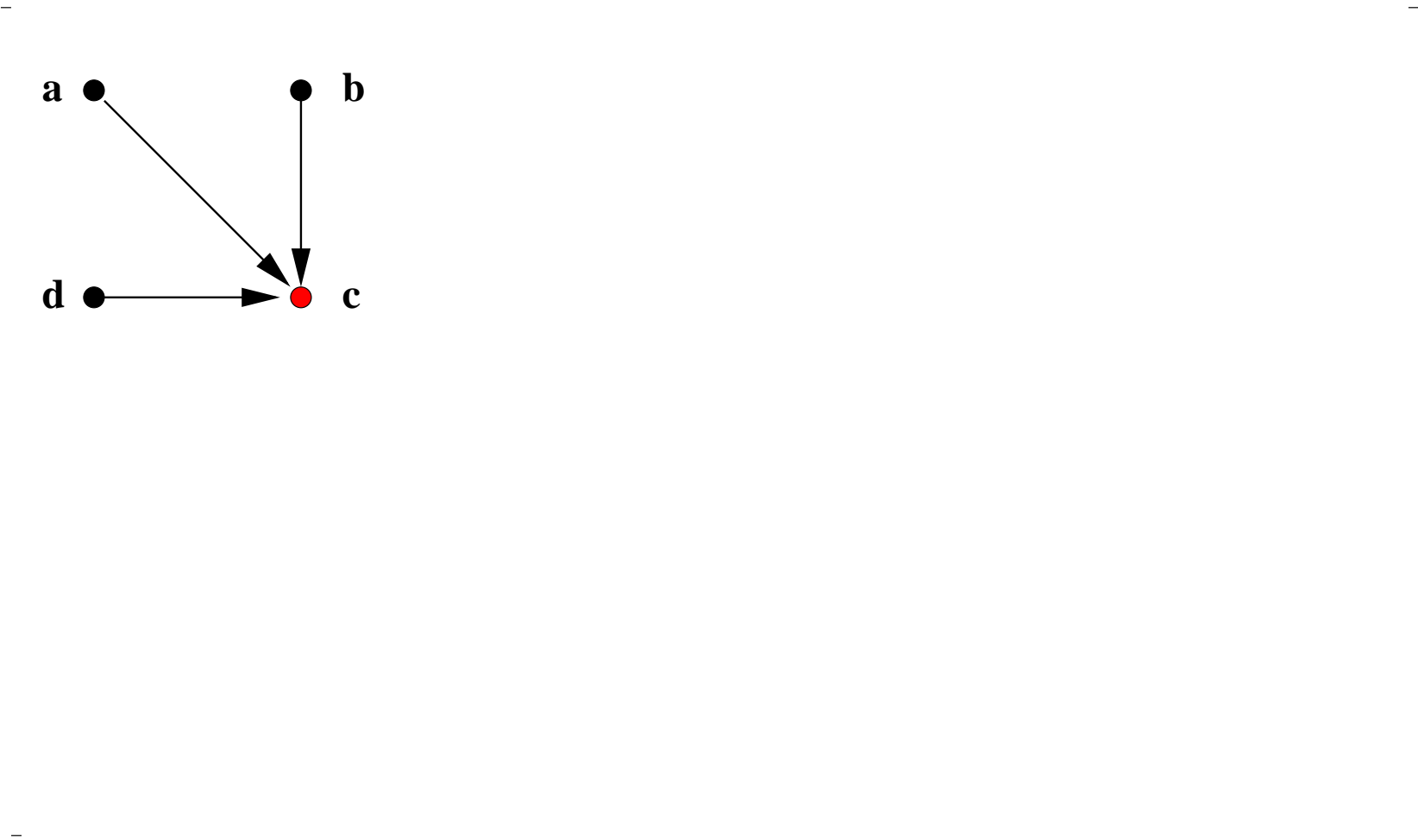


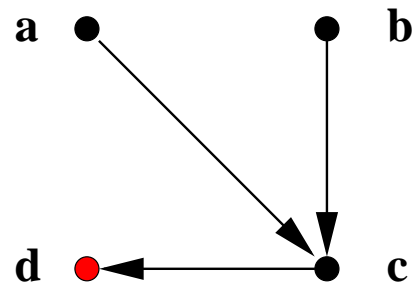
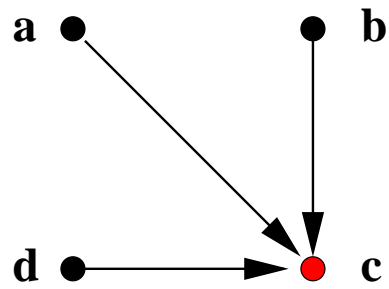


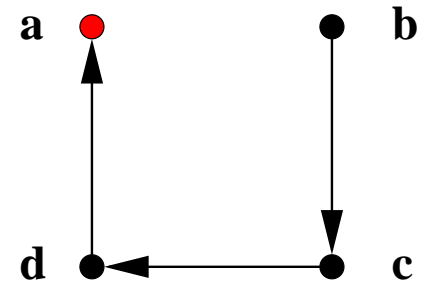
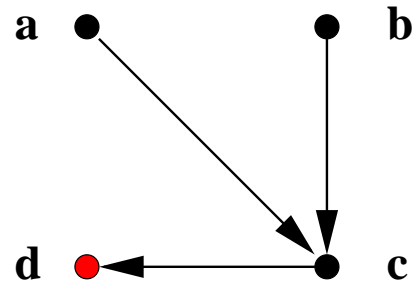
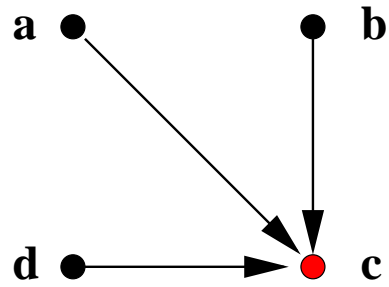


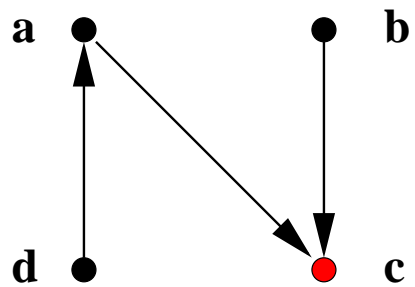
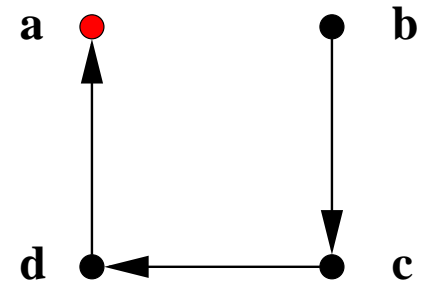
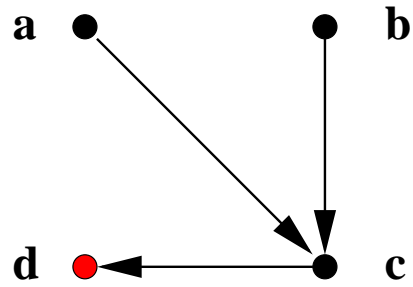
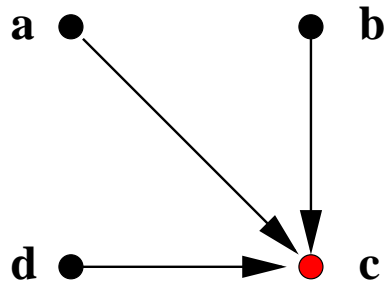
- The moves of \mathcal{M} are still instantaneous
- The traveling of messages between sites still takes no time
- The knowledge of the moves of \mathcal{M} is not instantaneous any more

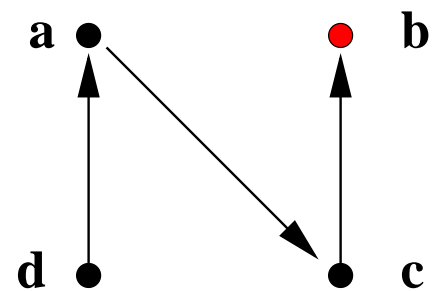
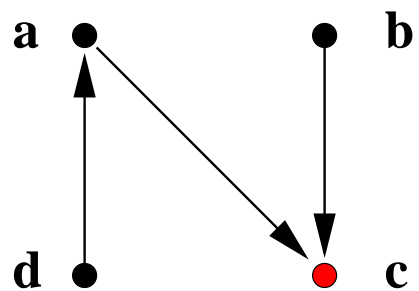
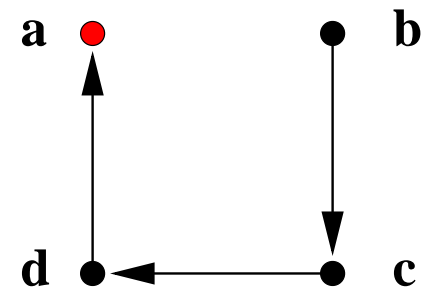
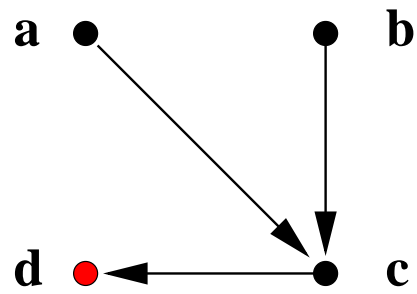
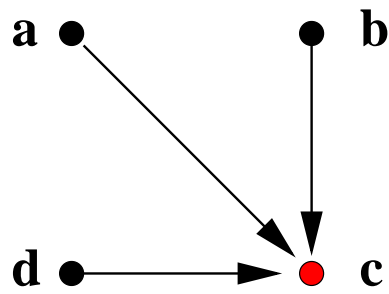
- When \mathcal{M} moves from site x to site y then
 - Agents of x and y **knows it immediately**
 - Agents of other sites are **not aware of the move**
 - They still sent their messages **where they believe \mathcal{M} is**
- A message arriving at a site which \mathcal{M} has left **can be forwarded**

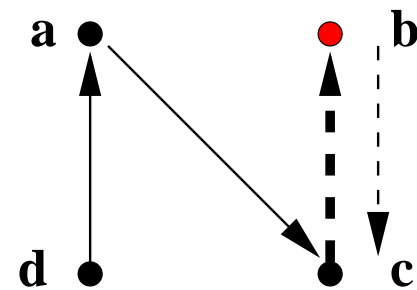
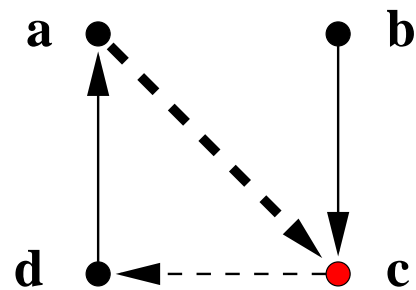
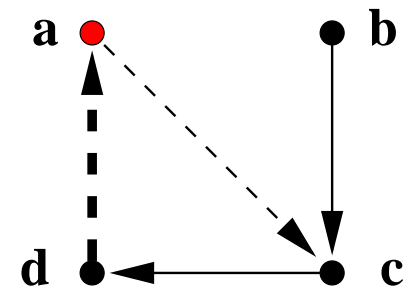
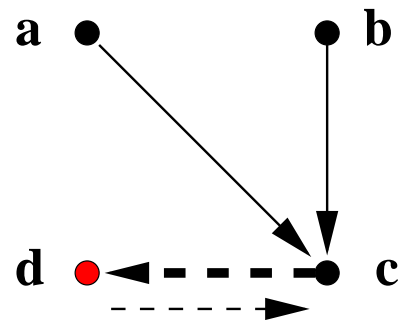
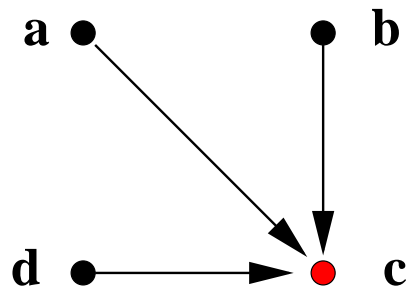


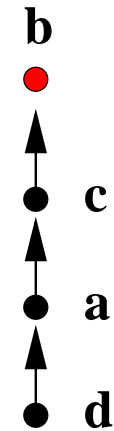
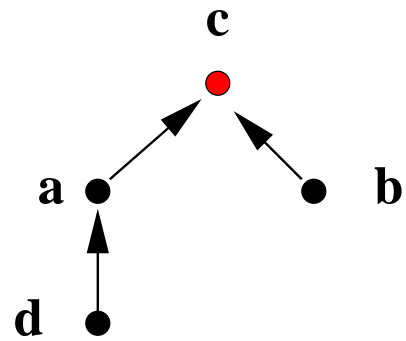
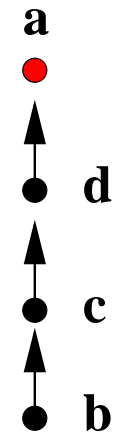
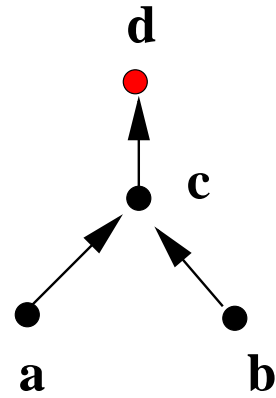
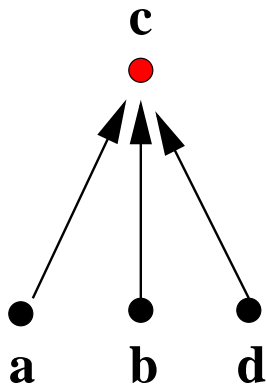


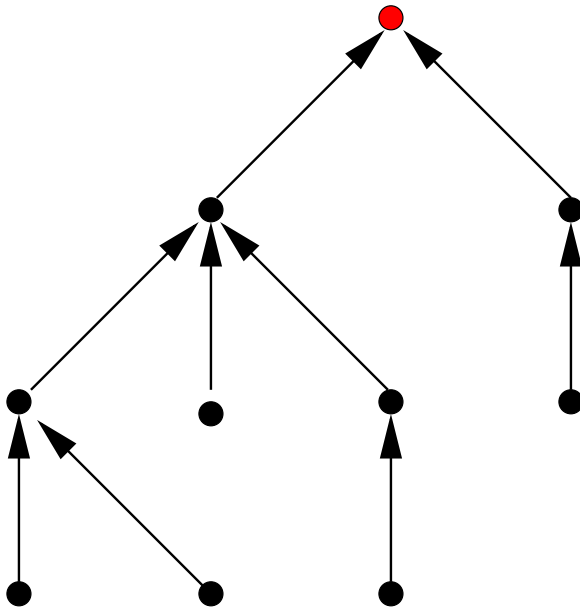




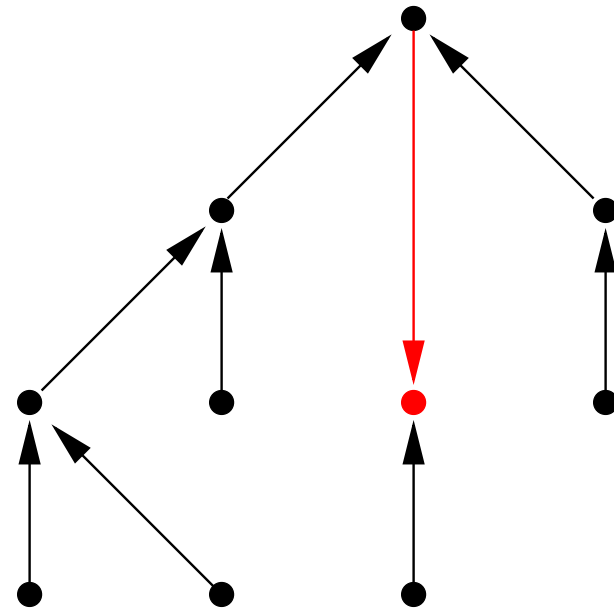
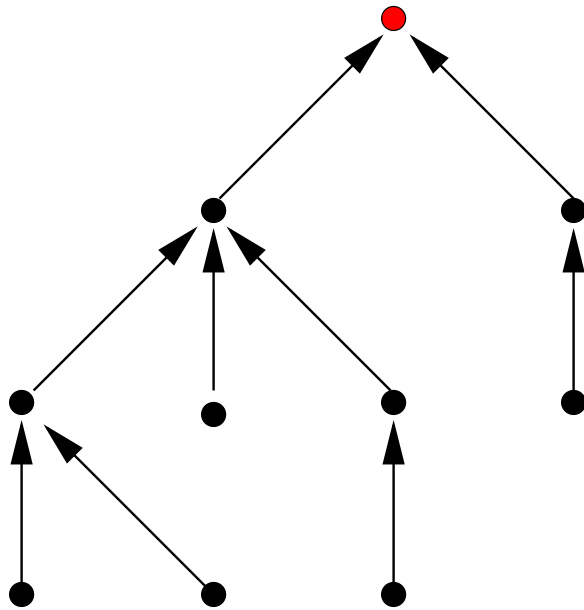


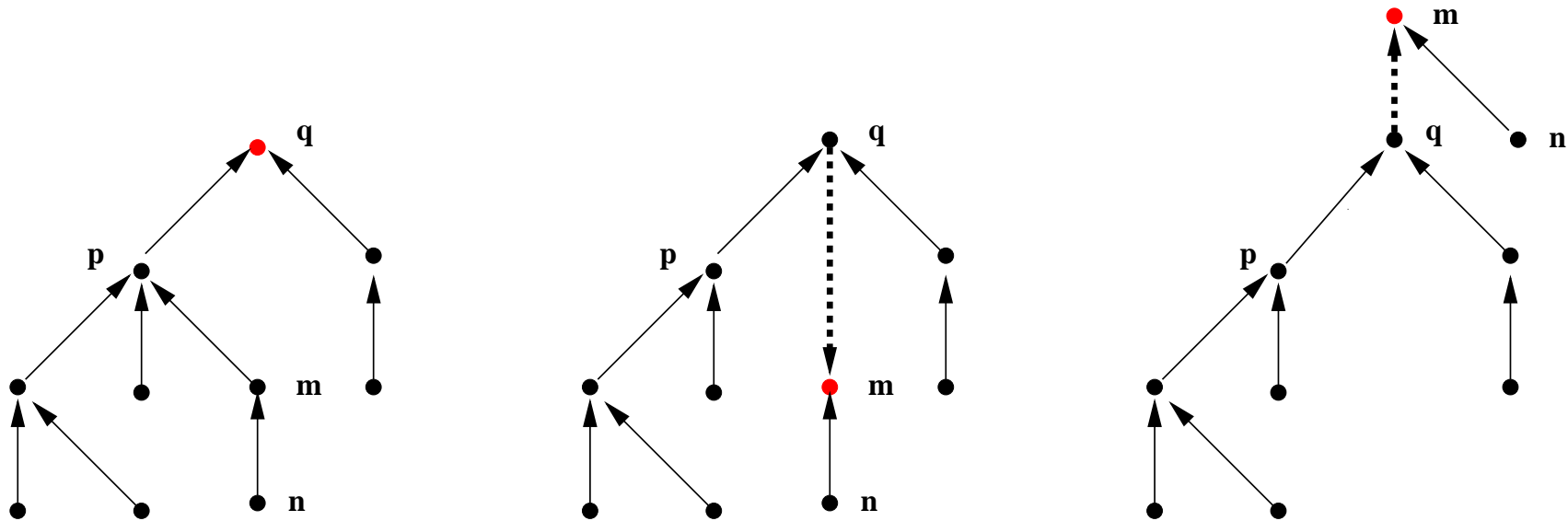






- The mobile \mathcal{M} is at the root of a tree





The mobile \mathcal{M} remains at the **root of a tree** (to be **proved** however)

sets: S
 M

axm0_1: $il \in S$

constant: il

- Constant il denotes the **initial location** of the Mobile

variables: l
 c
 p

inv0_1: $l \in S$

inv0_2: $c \in S \setminus \{l\} \rightarrow S$

inv0_3: $p \in M \rightarrow S$

- Variable l denotes the **actual location** of the Mobile
- Variable c denotes the **dynamic channel** structure
- Variable p denotes the **position of each message**

This invariant states that the **channel structure is a tree** with:

- **root**: l
- **parent function**: c

$$\text{inv0_4:} \quad \forall U \cdot U \subseteq c^{-1}[U] \Rightarrow U = \emptyset$$

```
init
   $l := il$ 
   $c := (S \setminus \{il\}) \times \{il\}$ 
   $p := \emptyset$ 
```

```
rcv_agt
  any  $s$  where
     $s \neq l$ 
  then
     $l := s$ 
     $c := (\{s\} \triangleleft c) \cup \{l \mapsto s\}$ 
  end
```

- This event describes the move of the Mobile from l to s
- The move of the mobile from l to s is supposed to be instantaneous

- Node s sends a message to the Mobile
- This message is stored locally

```
snd_msg  
  any  $s, m$  where  
     $s \in S$   
     $m \notin \text{dom}(p)$   
  then  
     $p(m) := s$   
  end
```

- Messages are either **delivered** or **forwarded**

```
dlv_msg
  any  $m$  where
     $m \in \text{dom}(p)$ 
     $p(m) = l$ 
  then
     $p := \{m\} \triangleleft p$ 
  end
```

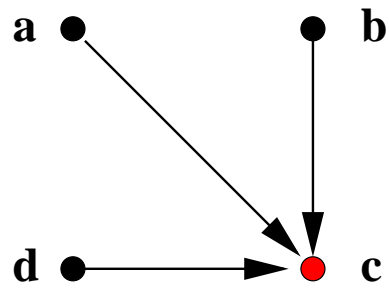
```
fwd_msg
  any  $m$  where
     $m \in \text{dom}(p)$ 
     $p(m) \neq l$ 
  then
     $p(m) := c(p(m))$ 
  end
```

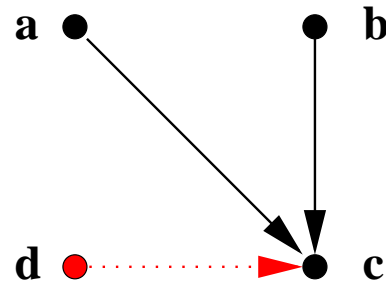
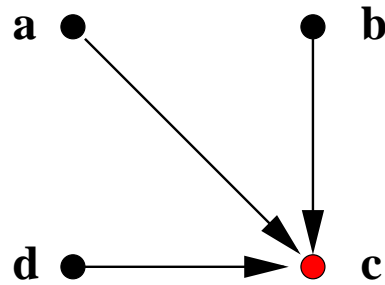
- When delivered, a message is **removed**

-
- The moves of \mathcal{M} are not completely instantaneous any more
 - The traveling of messages between sites still takes no time
 - The knowledge of the moves of \mathcal{M} is not instantaneous any more

- Agents of l do not know where \mathcal{M} is going
- Agents of other sites are not aware of the move
- Messages at l cannot be forwarded until l knows where \mathcal{M} is
- Messages at other sites can be forwarded (in general)

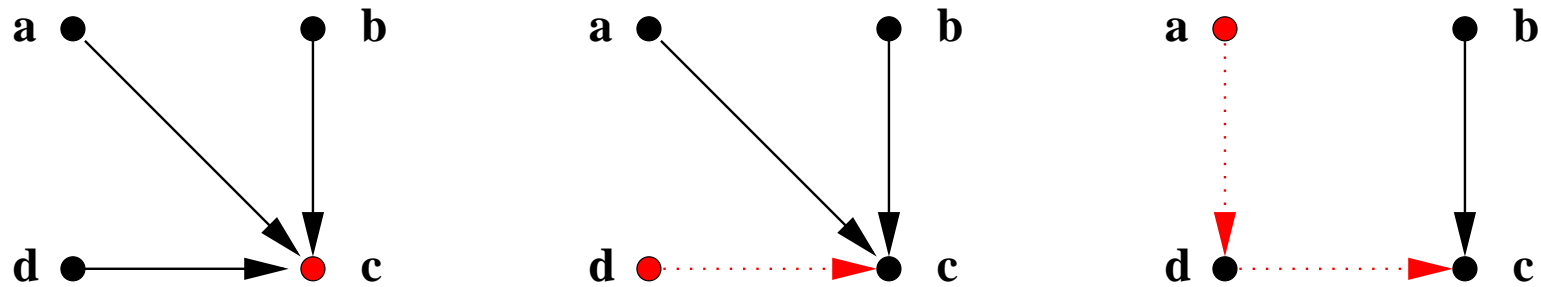
- It sends a “service message” to l to inform it about its new position
- Once l has received the “service message” it can forward again communication messages which were pending
- From now on, we have to distinguish:
 - communication messages (still instantaneous)
 - service messages (which take some time)





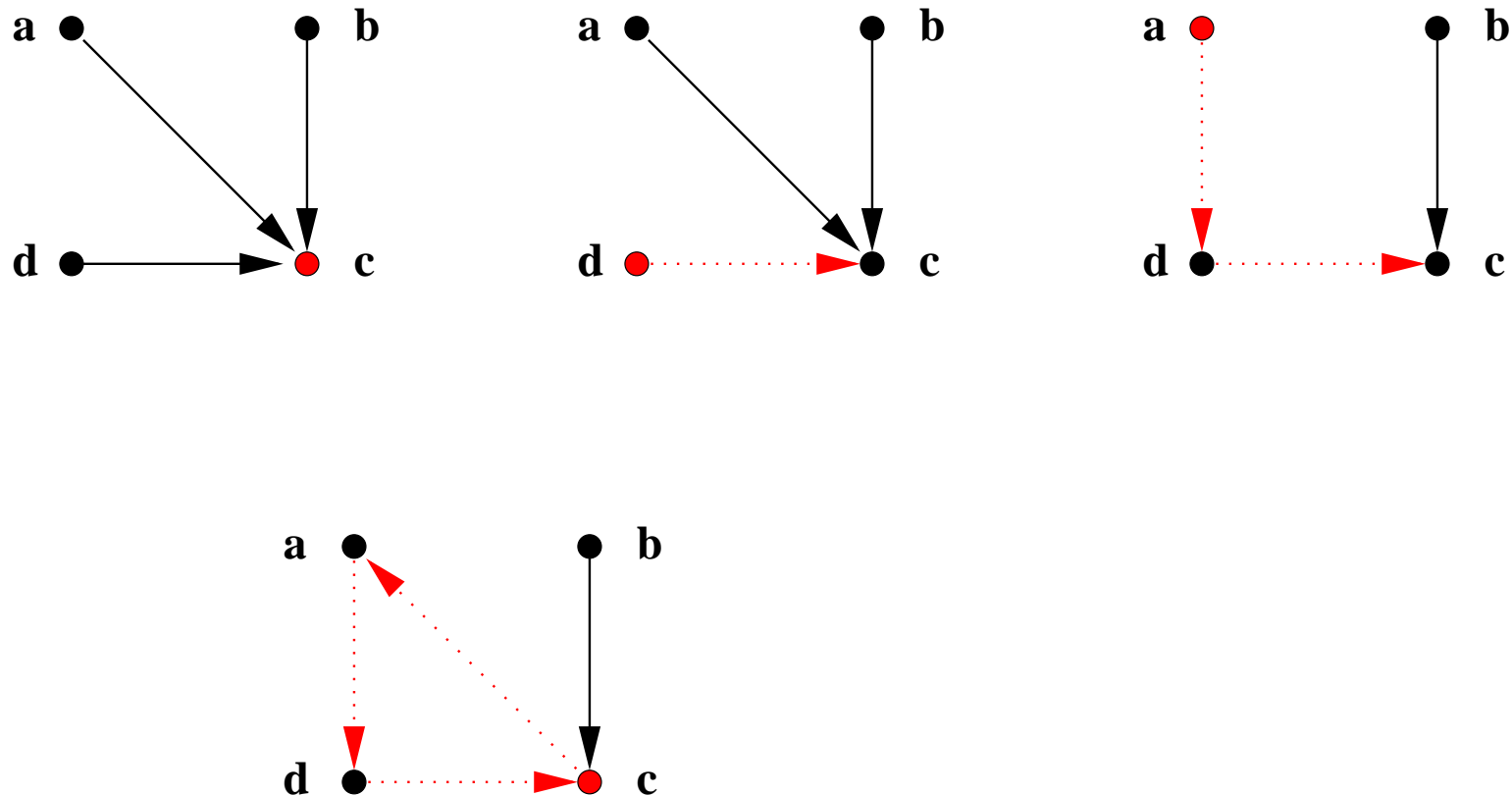
\mathcal{M} sends a service message to c : "I am now in d"

Site c suspend sending com. msg. until it knows where \mathcal{M} is



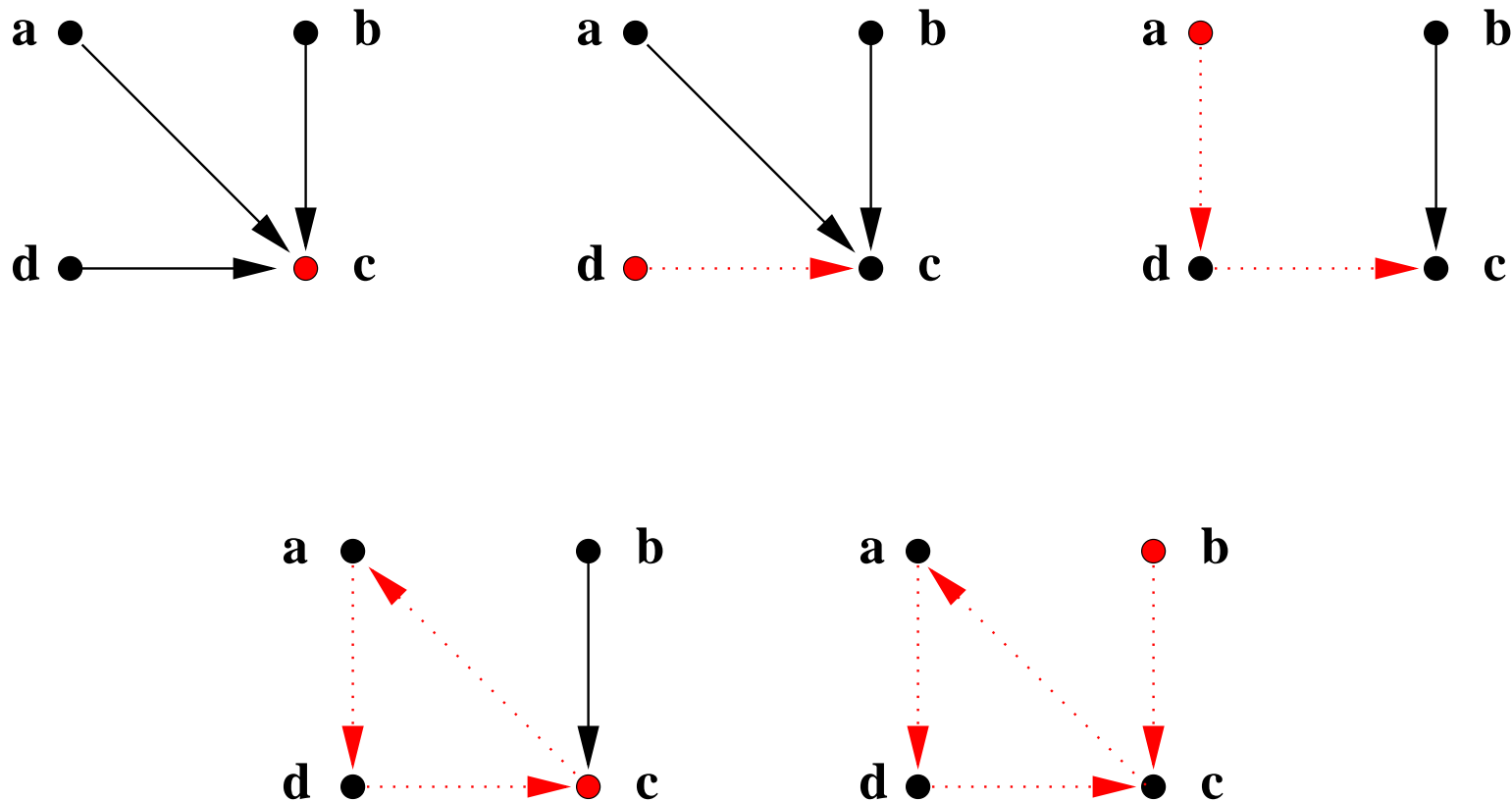
\mathcal{M} sends a service message to d : "I am now in a"

Site d suspend sending com. msg. until it knows where \mathcal{M} is



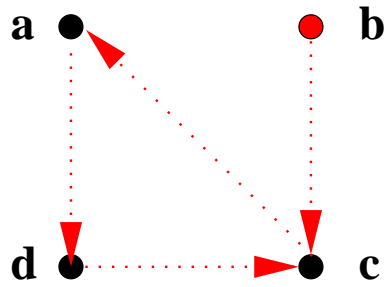
\mathcal{M} sends a service message to a : "I am now in c "

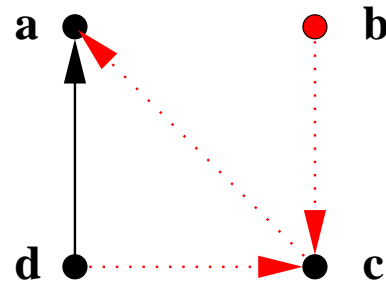
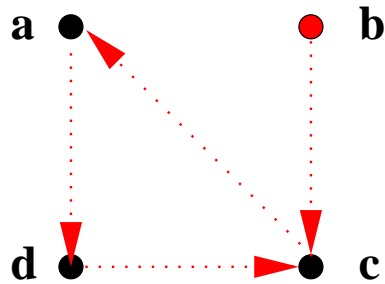
Site a suspend sending com. msg. until it knows where \mathcal{M} is



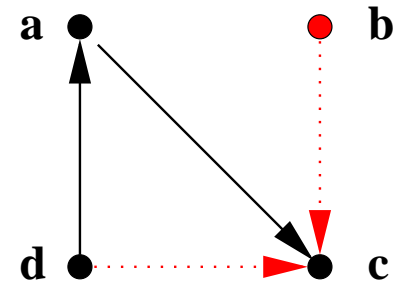
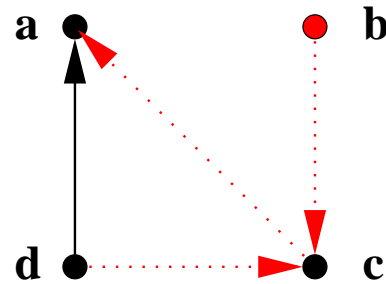
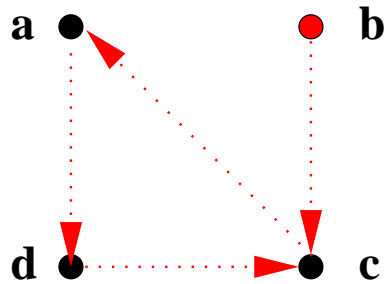
\mathcal{M} sends a service message to c : "I am now in b"

Site c suspend sending com. msg. until it knows where \mathcal{M} is

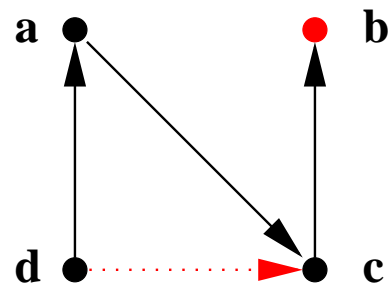
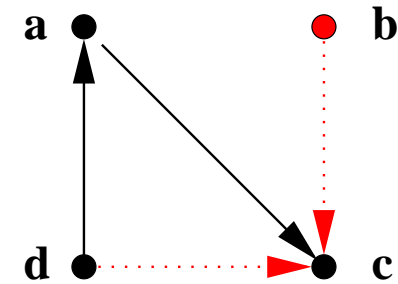
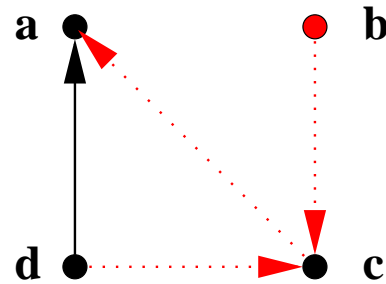
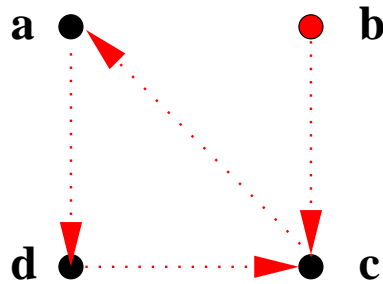




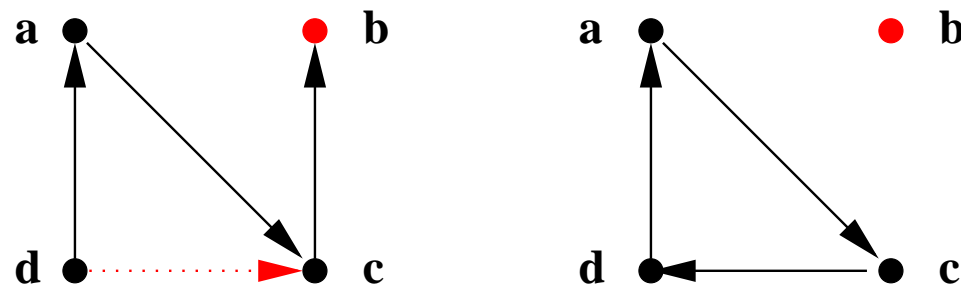
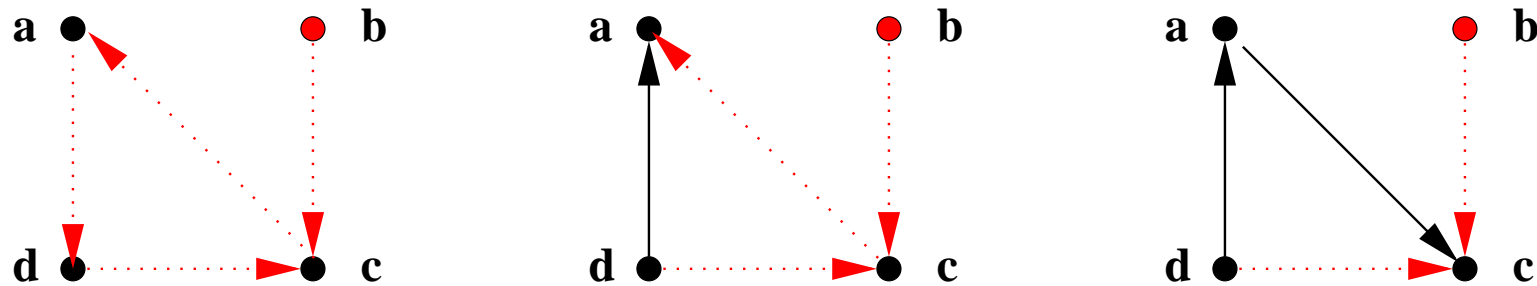
Site d believes \mathcal{M} is in a . It now forwards pending com. msg. to a



Site a believes \mathcal{M} is in c . It now forwards pending com. msg. to c

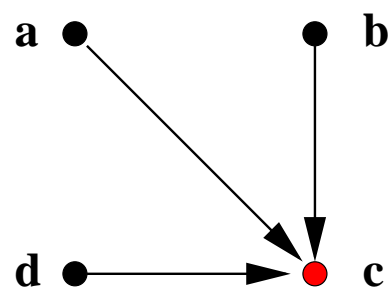


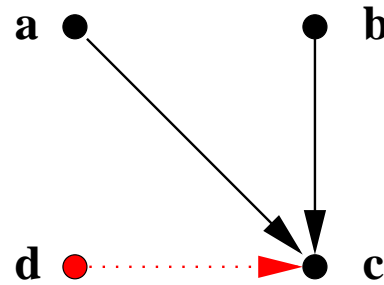
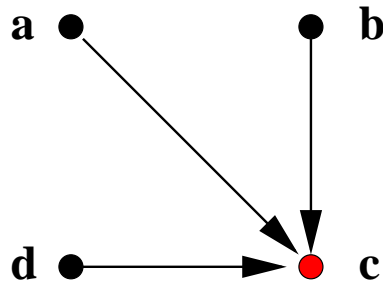
Site c believes \mathcal{M} is in b . It now forwards pending com. msg. to b



Site c believes \mathcal{M} is in d . It now forwards pending com. msg. to d
 The tree structure is destroyed: we have a **CYCLE**.

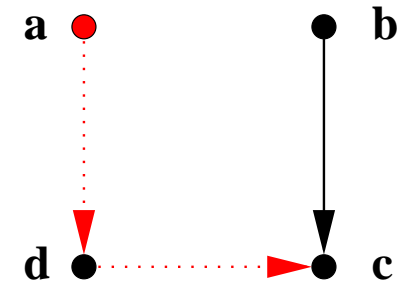
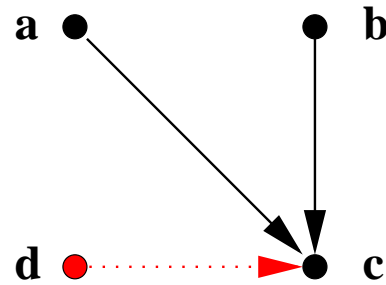
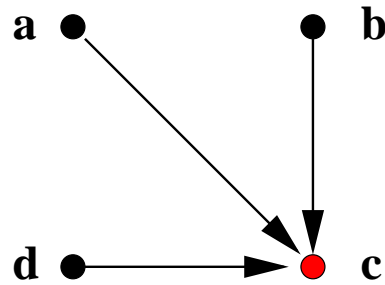
- The failure comes from the two srv. msg. arriving in the same place
- We must preclude this to happen
- We shall suppose that we have the following “magic” behavior
 - When \mathcal{M} sends a service message to site x
 - It is able to remove all other pending service messages whose destination is also x





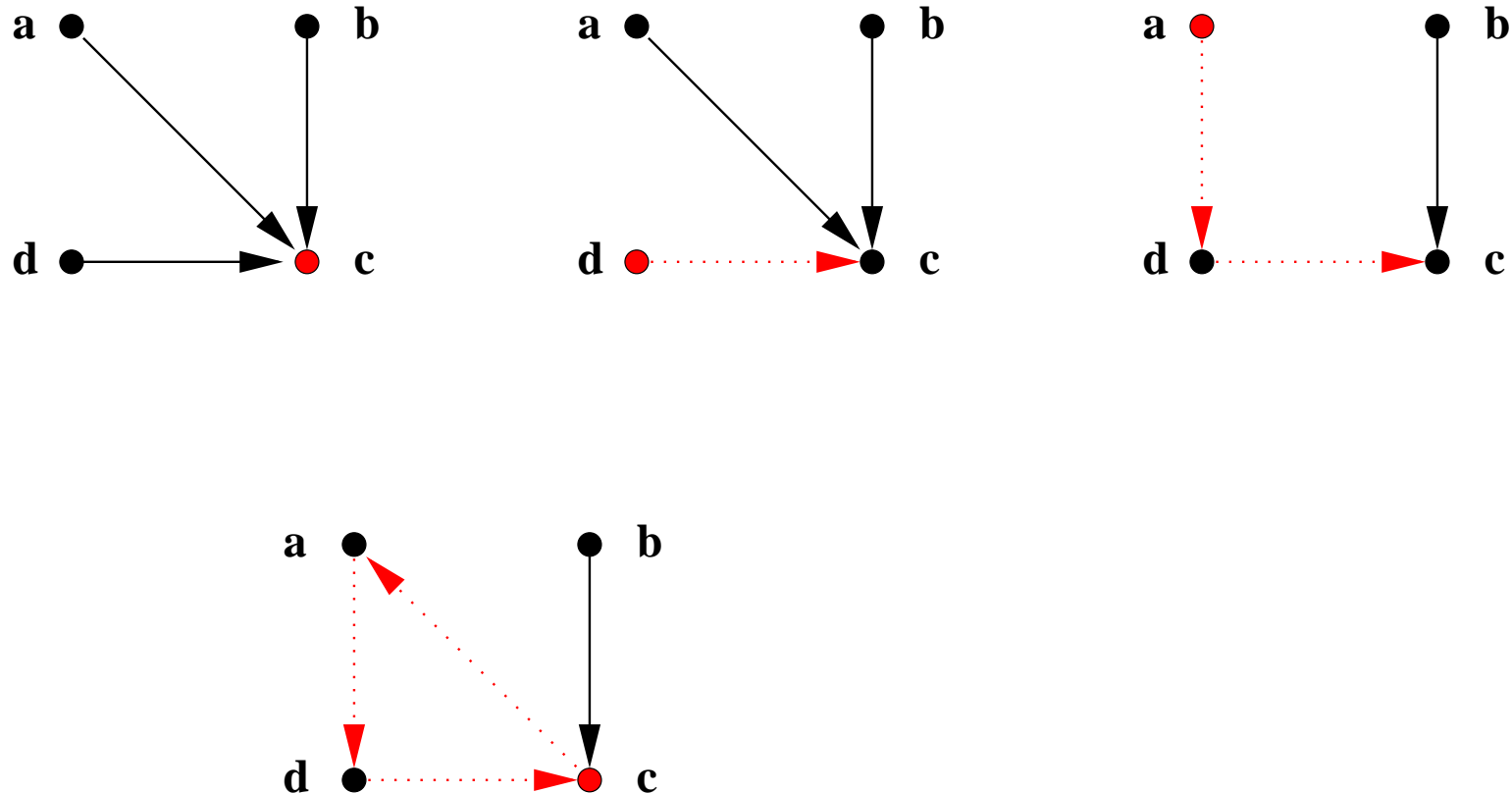
\mathcal{M} sends a service message to c : "I am now in d"

Site c suspend forwarding com. msg. until it believes where \mathcal{M} is



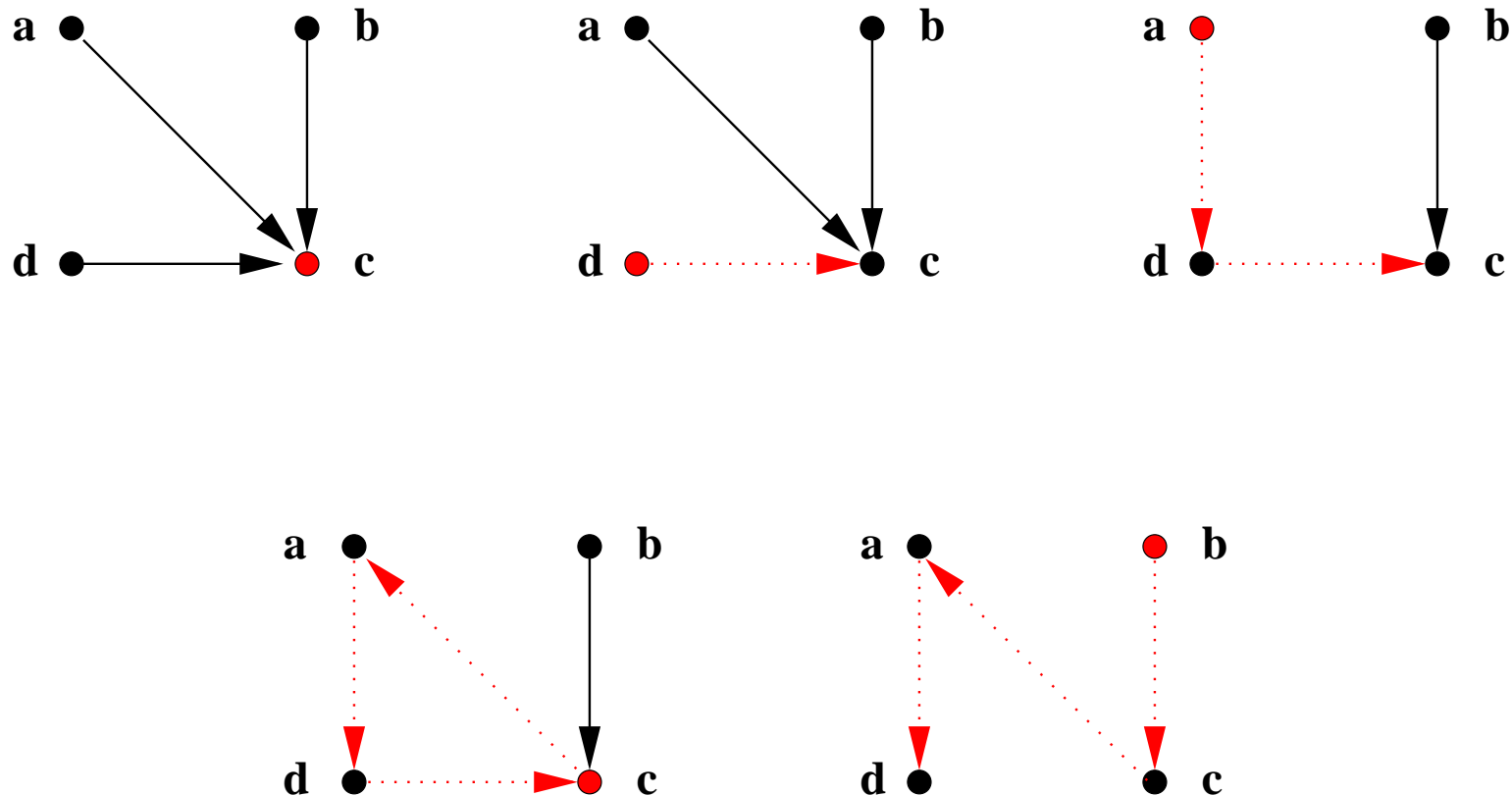
\mathcal{M} sends a service message to d : "I am now in a"

Site d suspend forwarding com. msg. until it believes where \mathcal{M} is



\mathcal{M} sends a service message to a : "I am now in c"

Site a suspend forwarding com. msg. until it believes where \mathcal{M} is



\mathcal{M} sends a service message to c : "I am now in b "

\mathcal{M} "magically" removes the other service message arriving to c

variables: l, p, d, a, da

inv1_1: $d \in S \setminus \{l\} \rightarrow S$

inv1_2: $a \in S \setminus \{l\} \rightarrow S$

inv1_3: $c = d \Leftarrow a$

inv1_4: $\text{dom}(a) = da \setminus \{l\}$

- Variable d denotes the new dynamic tree structure
- Variable a denotes the service message channel.
- **inv1_3** denotes the link between c and the concrete d and a
- Variable da denotes the set of sites expecting a service message
- Such nodes cannot forward a message

$$\text{inv1_2: } a \in S \setminus \{l\} \rightarrow S$$

- $s1 \mapsto s2$ in a means a message from $s2$ (new site) to $s1$ (old site)
- At most one service message is in transit to site $s1$ (a is a function)
- This magic behavior is fundamental

dlv_msg

any m **where**

$m \in \text{dom}(p)$

$p(m) \notin da$

$p(m) = l$

then

$p := \{m\} \triangleleft p$

end

fwd_msg

any m **where**

$m \in \text{dom}(p)$

$p(m) \notin da$

$p(m) \neq l$

then

$p(m) := d(p(m))$

end

- The guards are now **local**
- We can later **data-refine** da with a local **boolean variable**


```

leave_agt
  when
     $l \notin da$ 
  then
     $da := da \cup \{l\}$ 
  end

```

```

rcv_agt
  any  $s$  where
     $s \neq l$ 
     $l \in da$ 
  then
     $l := s$ 
     $a := (\{s\} \triangleleft a) \triangleleft \{l \mapsto s\}$ 
     $d := \{s\} \triangleleft d$ 
     $da := da \setminus \{s\}$ 
  end

```

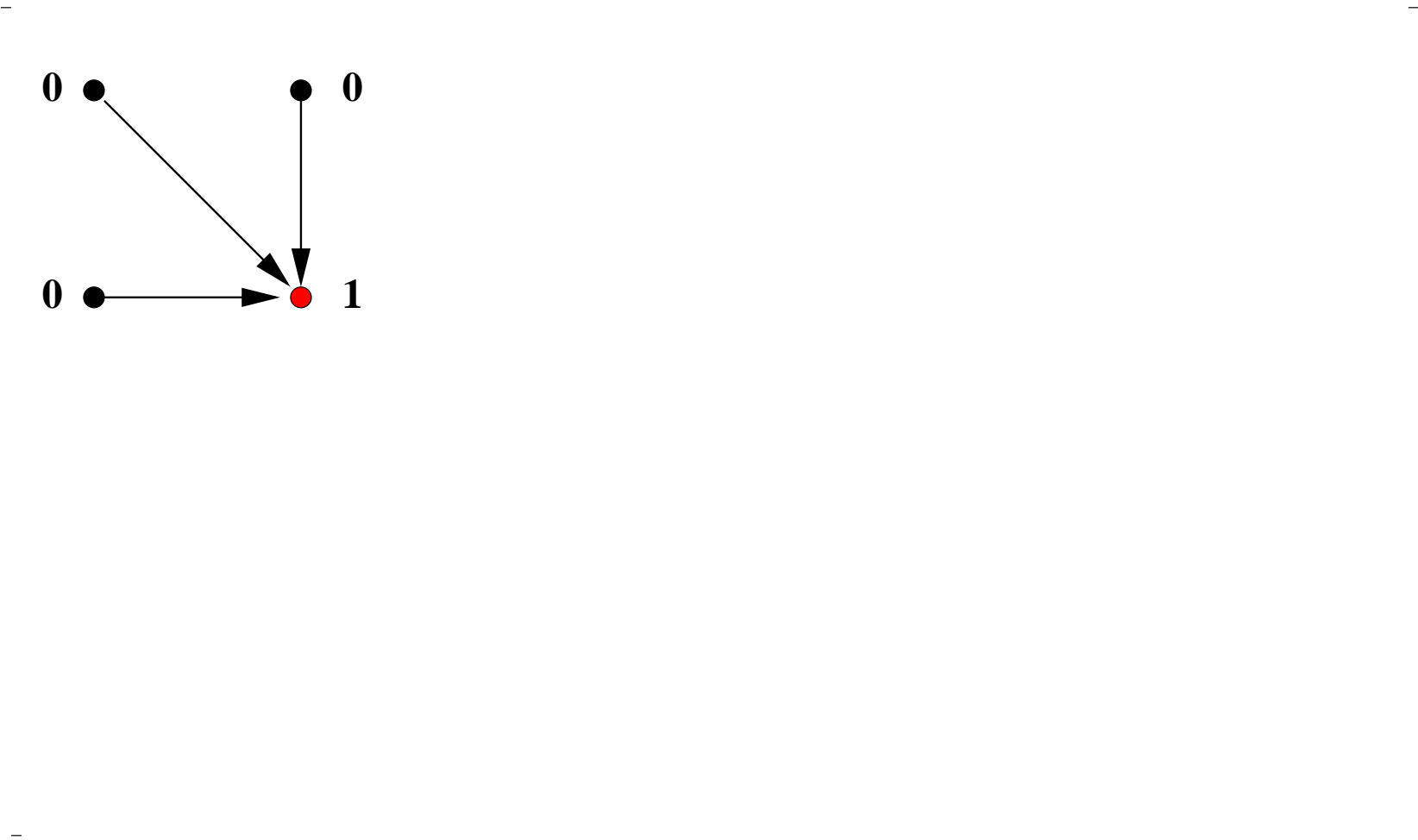
- Event **leave_agt** is a **new event** where the set **da** is extended.
- In event **rcv_agt**, the new site location s is **removed from da**.
- A previous service message to l is **removed**.
- When l is in **da**, it means that the mobile **\mathcal{M}** is in transit

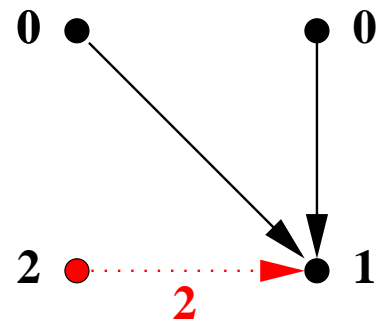
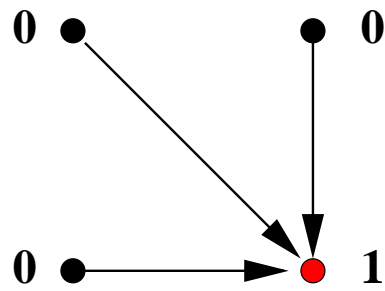
```
rcv_srv
  any  $s$  where
     $s \in \text{dom}(a)$ 
     $s \neq l$ 
  then
     $d(s) := a(s)$ 
     $a := \{s\} \triangleleft a$ 
     $da := da \setminus \{s\}$ 
  end
```

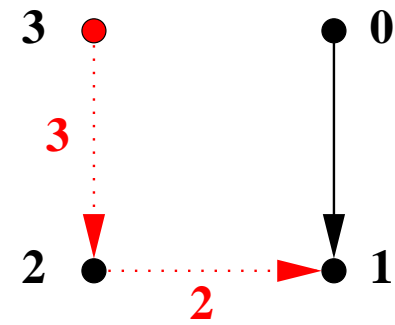
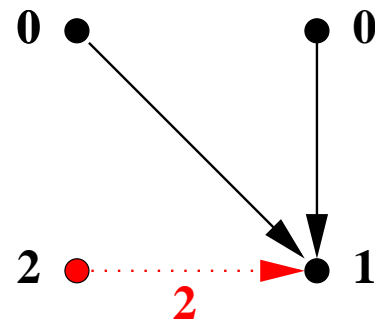
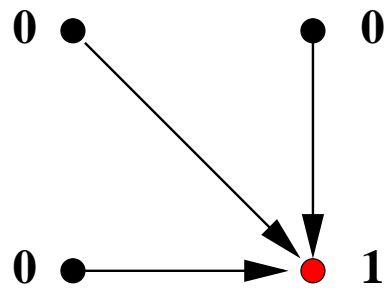
- This is a **new event**
- It corresponds to the **arrival of the service message**

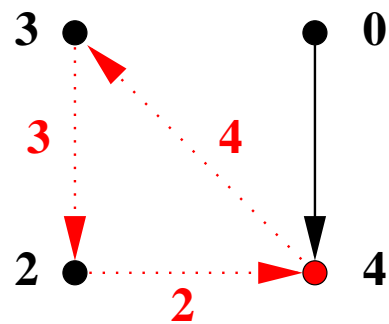
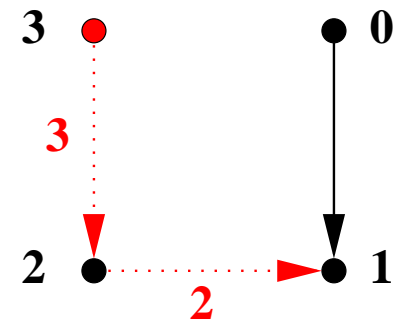
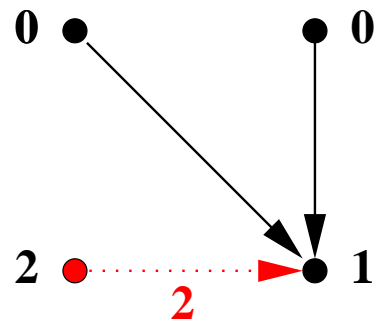
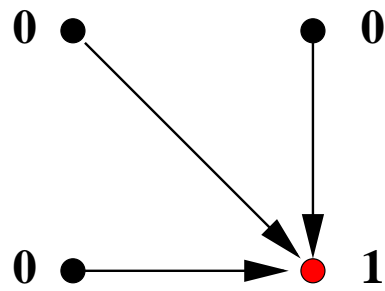
- Magic behavior when sending a new service message to x :
 - Pending service messages to x are removed
- The mobile \mathcal{M} travels with a logical clock
- Each site has a last time counter
- This counter records the “time” of the last visit of \mathcal{M}

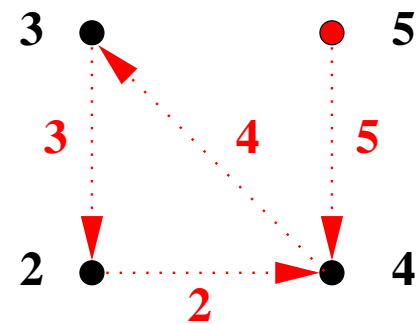
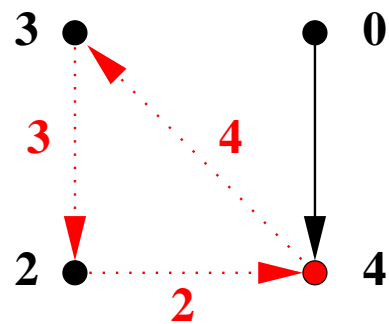
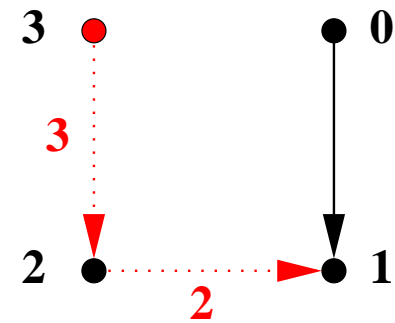
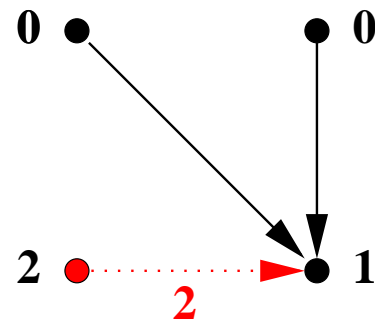
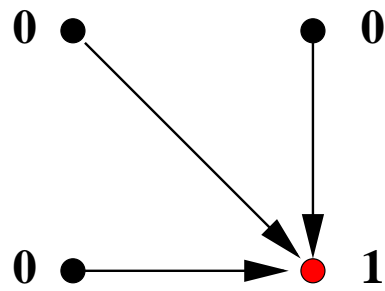
- When \mathcal{M} arrives at a site y
 - it increments its logical clock
 - it stores its incremented clock in the last time counter of y
 - it sends a new service message to its previous location x
- The srv. msg. from y to x is stamped with the new clock value
- When a service message arrives at a site x , it is accepted
 - only if its stamp value is greater than the time counter of x
 - the last time counter takes the value of the stamp

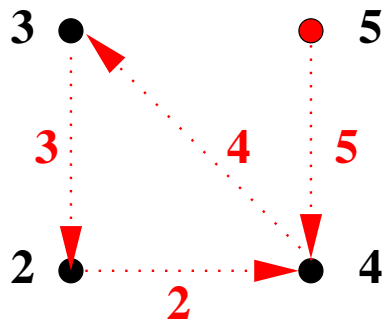


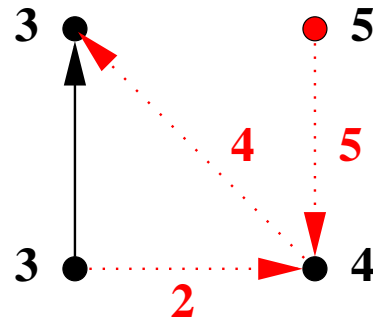
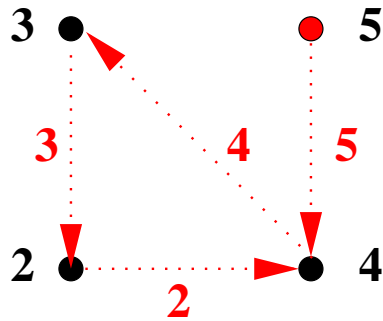




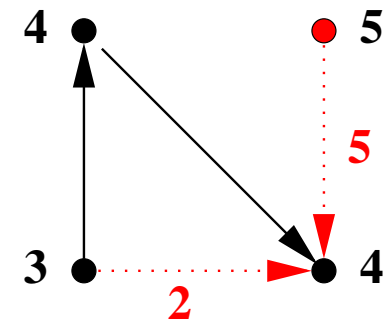
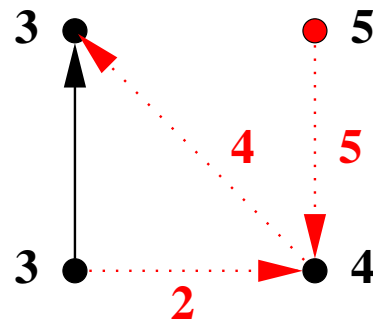
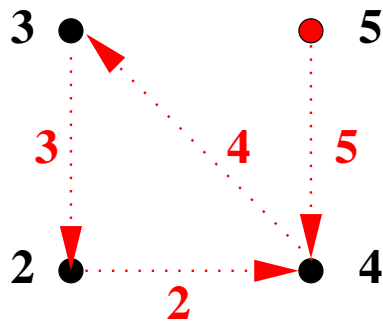




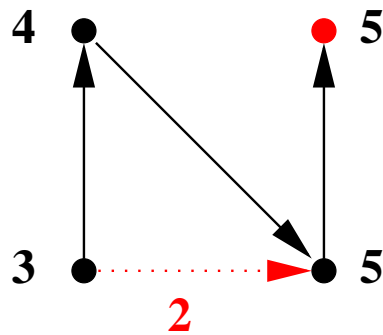
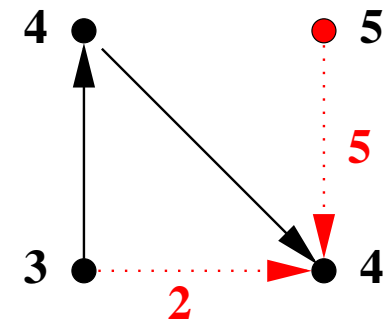
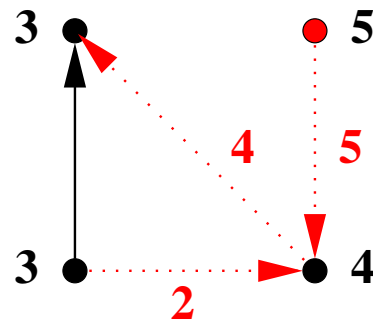
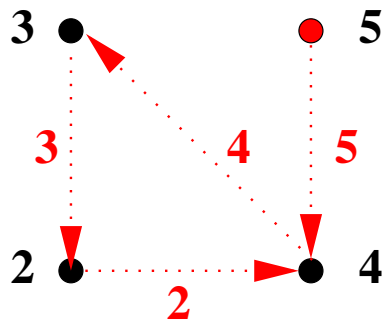




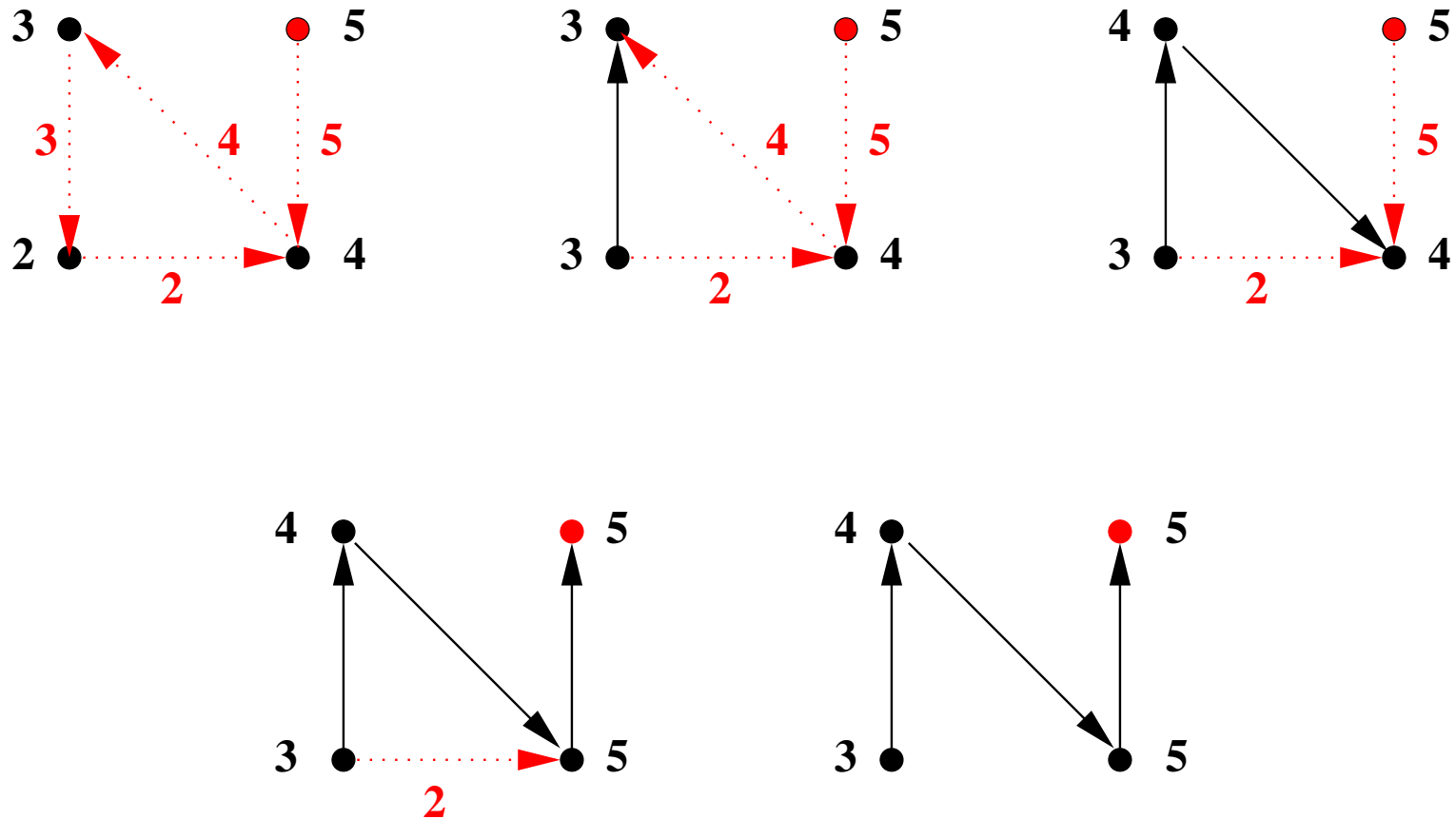
- It is **accepted**



- It is **accepted**



- It is **accepted**



- It is **rejected**

- Suppose:
 - $s1$ has emitted a service msg. to s at time 3
 - $s2$ has emitted a service msg. to s at time 5
 - $s3$ has emitted a service msg. to s at time 9
- This will be “recorded” in the **refined service channel** as follows:

$$s \mapsto \{3 \mapsto s1, 5 \mapsto s2, 9 \mapsto s3\}$$

- In the **abstract service channel** we had: $s \mapsto s3$

variables: $l, p, d, da,$
 k, t, b

inv2_1: $k \in \mathbb{N}$

inv2_2: $t \in S \rightarrow \mathbb{N}$

inv2_3: $b \in S \rightarrow (\mathbb{N} \leftrightarrow S)$

- Variable k is the **clock** taken by the Mobile when it travels
- Variable t denotes the **time of the last visit** of the Mobile to a site
- Variable b is the new service channel, **it data-refines variable a**

- An **abstract** service message is the **most recent concrete one**

$$\text{inv2_4: } \forall s . \left(\begin{array}{l} s \in \text{dom}(a) \\ \Rightarrow \\ \text{dom}(b(s)) \neq \emptyset \\ a(s) = b(s)(\max(\text{dom}(b(s)))) \end{array} \right)$$

$$\text{inv2_5: } \forall s \cdot \left(\begin{array}{l} s \in S \\ \text{dom}(b(s)) \neq \emptyset \\ t(s) < \max(\text{dom}(b(s))) \\ \Rightarrow \\ s \in \text{dom}(a) \end{array} \right)$$

- This technical invariant will help us proving **guard strengthening** for event `rcv_srv`

inv2_6: $\forall s \cdot (s \in S \wedge \text{dom}(b(s)) \neq \emptyset \Rightarrow \max(\text{dom}(b(s))) \leq k)$

inv2_7: $t(l) = k$

inv2_8: $\forall s \cdot (s \in S \setminus \{l\} \Rightarrow t(s) \leq k)$

- The only **service message stamp** to a site s which is strictly **greater than the time of last visit** to that site s is the **maximum one**.

$$\text{inv2_9: } \forall s, n \cdot \left(\begin{array}{l} s \in S \\ n \in \text{dom}(b(s)) \\ t(s) < n \\ \Rightarrow \\ n = \max(\text{dom}(b(s))) \end{array} \right)$$

- Sending the service message with the **time stamp $k + 1$**

```
(abstract-)rcv_agt
  any  $s$  where
     $s \in S \setminus \{l\}$ 
     $l \in da$ 
  then
     $l := s$ 
     $a(l) := s$ 
     $d := \{s\} \triangleleft d$ 
     $da := da \setminus \{s\}$ 
  end
```

```
(concrete-)rcv_agt
  any  $s$  where
     $s \in S \setminus \{l\}$ 
     $l \in da$ 
  then
     $l := s$ 
     $t(s) := k + 1$ 
     $k := k + 1$ 
     $b(l)(k + 1) := s$ 
     $d := \{s\} \triangleleft d$ 
     $da := da \setminus \{s\}$ 
  end
```

```

    (abstract-)rcv_srv
    any  $s$  where
         $s \in \text{dom}(a)$ 
         $s \neq l$ 
    then
         $d(s) := a(s)$ 
         $a := \{s\} \triangleleft a$ 
         $da := da \setminus \{s\}$ 
    end

```

$$\forall s . \left(\begin{array}{l} s \in S \\ \text{dom}(b(s)) \neq \emptyset \\ t(s) < \max(\text{dom}(b(s))) \\ \Rightarrow \\ s \in \text{dom}(a) \end{array} \right)$$

```

    (concrete-)rcv_srv
    any  $s, n$  where
         $s \in S$ 
         $n \in \text{dom}(b(s))$ 
         $t(s) < n$ 
    then
         $d(s) := b(s)(n)$ 
         $t(s) := n$ 
         $da := da \setminus \{s\}$ 
    end

```

$$\forall s, n . \left(\begin{array}{l} s \in S \\ n \in \text{dom}(b(s)) \\ t(s) < n \\ \Rightarrow \\ n = \max(\text{dom}(b(s))) \end{array} \right)$$

Initial Model	11	0
1st Reft.	23	2
2nd Reft.	70	7
Total	104	9